update on perf_events

Stephane Eranian
CERN workshop, November 2013

Agenda

- improve multiplexing accuracy
- load/store sampling
- event grouping with perf
- multi-event sampling
- count aggregations
- Intel Haswell support
- uncore update
- hw bugs

Improved multiplexing scaling

- Available in Linux 3.11
- Multiplexing triggered on timer ticks
  - tickless kernel: no timer tick when idle = no multiplexing
  - events may happen while core is idle (e.g., uncore events)
- switch to per-cpu hrtimer for trigger
  - wake-up from idle to service timer
  - adjustable period in ms per PMU instance via `sysfs`
  - Example: **`echo 10 >/sys/devices/cpu/perf_event_mux_interval_ms`**

Example: idle system, ref-cycles works on 1 counter only:

```
# perf stat -e ref-cycles,ref-cycles -a sleep 10
      5 825 973 800 ref-cycles    [50,01%]
      5 980 094 548 ref-cycles    [49,99%]
```

Memory access sampling

- Available in Linux 3.10
  - requires HW support (NHM load only, WSM, SNB, IVB, HSW)
  - PPC7/PPC8 support in progress
- load/store sampling with Precise Event-Based Sampling (PEBS)
  - load: instr & data addr, instr latency, data source
  - store: instr & data addr, limited data source
  - data source abstracted: mem lvl, tlb lvl, snoop, lock
  - warning: instruction latency from dispatch (not just miss latency)
- perf tool support
  - `perf mem`: new wrapper command (record, report)
  - use `perf mem -D` for raw dump for post-processing

Google

Memory access sampling example

```
$ perf mem -t load rec test

$ perf mem -t load rep --stdio
# Sort order   : local_weight,mem,sym,dso,symbol_daddr,dso_daddr,snoop,tlb,locked
# OV    Smpl  Weight Mem                          Sym        Obj         Data Sym        Data
# ..    ....  ...... ...    ........................  .......  ..............  ..........
 1.72%   92    1386 L3 hit [.] acquire.constprop.1  struct2  [.] object+0x18    struct2
 1.37%   73    1387 L3 hit [.] release.constprop.0  struct2  [.] object+0x18    struct2
 1.07%   57    1388 L3 hit [.] acquire.constprop.1  struct2  [.] object+0x18    struct2

$ perf mem -t load rep --sort=mem --stdio
# Samples: 23K of event 'cpu/mem-loads/pp'
# Sort order   : mem
#       OV       Samples             Memory access
# ........  ............  ......................
    97.95%          9915  L3 hit
     2.04%         13320  L1 hit
     0.01%            10  LFB hit
     0.00%             1  Local RAM hit
     0.00%             3  L2 hit
     0.00%             1  Uncached hit
```

Counting with event groups

- Available from Linux-3.8
- enforce event grouping from perf cmdline
  - events in group are always measured together
  - group cannot have more events than counters (ignoring constraints)
  - events must be compatible with each other (no counter conflict)
- supported by perf record as well

```
$ perf stat -e "{cycles,instructions},{branches,instructions}" noploop
        3588967528 cycles
        3585528414 instructions
        3584888150 branches
        3585528414 instructions
```

Multi-event profiles (1)

- available in Linux 3.8
- multi-event sampling not new, profile output is
  - single merge profile vs. single-event profiles
  - use `--group` option on `perf record`, `report`, `annotate`
- can be combined with explicit event grouping

Google

Multi-event profiles (2)

```
$ perf record --group -e cycles,instructions noploop 2
$ perf report --group --stdio

# Samples: 16K of event 'anon group { cycles, instructions }'
# Event count (approx.): 9346466161
#         Overhead  Command      Shared Object                      Symbol
# ................  .......  .................  ........................
    99.95%  99.98%  noploop  noploop            [.] noploop
     0.02%   0.01%  noploop  [kernel.kallsyms]  [k] __slab_free
     0.01%   0.00%  noploop  ld-2.15.so         [.] _dl_relocate_object

$ perf annotate --group --stdio
Percent         |      Source code & Disassembly of noploop
------------------------------------------------------------
                :      0000000000400629 <noploop>:
   0.00    0.00 :        400629:       push   %rbp
   0.00    0.00 :        40062a:       mov    %rsp,%rbp
 100.00  100.00 :        40062d:       jmp    40062d <noploop+0x4>
```

Remote collection/local analysis with perf

```
$ ssh root@server “perf record -o perf.data -a sleep 10”
$ ssh server “perf archive”

$ scp server:perf.data /tmp

$ scp server:perf.data.bz2 /tmp/perf.data.bz2
$ tar -C ~/.debug -jxf /tmp/perf.data.bz2

$ perf report -i /tmp/perf.data
```

Energy consumption counter support

- Planned for 3.13
- Intel Running Average Power Limit (RAPL) counters
  - power limiting, energy consumption in Joules
  - available in SNB, IVB, HSW
  - consumption already reported by `turbostat` tool
- Integrated with `perf stat`
  - system-wide mode, counting only, package-level consumption
  - new events: `power/energy-cores/, power/energy-pkg, power/energy-dram/`

```
# perf stat -a -e power/energy-cores/,power/energy-pkg/ -I 1000 sleep 10
#           time             counts   unit events
     1.000119482               7.72 Joules power/energy-cores/
     1.000119482              12.67 Joules power/energy-pkg/
```

Counts aggregation per core/per socket

- available with Linux 3.9
- help with core or socket workload imbalance detection
  - system-wide mode only
  - use: `--per-core` or `--per-socket` options
  - can be combined with delta printing of counts

Example: Intel Core I5

```
# perf stat -a --per-core -e cycles -I 1000 sleep 100
#           time core         cpus             counts unit events
     1.000113959 S0-C0            2      3 289 919 439      cycles
     1.000113959 S0-C1            2          5 357 472      cycles
     2.000527224 S0-C0            2      3 290 758 334      cycles
     2.000527224 S0-C1            2          5 762 741      cycles
```

Haswell PMU new features: counting TSX events

- available in Linux 3.12
- Transactional Memory (TSX) support
  - `in_tx`: count only when inside a transactional region (generic counters)
  - `in_tx_cp`: do not count in aborted transaction (generic_counters)
  - TSX related events: `HLE_*, RTM_*, TX_MEM_*`
  - generic TSX events in `/sys/devices/cpu/events/{el, tx}-*`

```
$ perf stat -e cpu/event=0x3c,in_tx=1/,cpu/cpu-cycles,in_tx_cp=1/,cycles nloop
1
noploop for 1 seconds
                 0 cpu/event=0x3c,in_tx=1/
     3 626 028 554 cpu/cpu-cycles,in_tx_cp=1/
     3 626 028 554 cpu/cpu-cycles/
       1,000981312 seconds time elapsed
```

Haswell new PMU features: sampling TSX events

- Available in Linux 3.12
  - use `gcc-4.8` for HLE, RTM support
- Any interrupt aborts a transaction
- PEBS TSX infos
  - `in_tx, in_tx_cp, any` filters are not supported
  - events with PEBS + info: `HLE_RETIRED:ABORTED, RTM_RETIRED:ABORTED`
  - TSX info: reason for abort, cycles spent in tx

```
$ perf record --transaction -e cpu/el-abort/pp hle_test
$ perf report --stdio --sort=comm,dso,sym,transaction
# Overhead  Command  Obj    Symbol            Transaction
# ........  .......  ..... ................  .................
    99.83%      hle  hle    [.] do_loop_hle   EL SYNC
     0.14%      hle  hle    [.] spin          EL ASYNC CAP-READ
     0.03%      hle  hle    [.] do_loop_hle   EL NEITHER
```

Haswell PEBS improvements: Eventing IP

- Available in Linux 3.11
- PEBS EventingIP
  - return address of sampled instruction
  - eliminates PEBS off-by-1 skid (because IP captured at retirement)
  - new `pebs->eventing_ip` field
  - off-by-1 IP still avail, useful for branch sampling
- EventIP with `precise=2`, off-by-1 IP with `precise=1`
  - LBR not used to correct off-by-1 anymore

```
$ perf record -e cpu/event=0xc4,umask=0x2/pp noploop 2 (BR_INST_RETIRED:NEAR_CALL)
```

Haswell PEBS improvements: Data Linear Address

- Available in Linux 3.10
- PEBS Data Linear Address (DLA)
  - capture data address for all PEBS memory events (see backup slides)
  - can capture data address for specific cache events (loads/stores)
- no integration with `perf report/mem yet`!
  - only raw dump possible

Sampling on `MEM_LOAD_UOPS_L3_MISS_RETIRED:LOCAL_DRAM` with PEBS + DLA:

```
$ perf record -d -e cpu/event=0xd3,umask=0x01/pp triad
$ perf report -D | fgrep SAMPLE
PERF_RECORD_SAMPLE IP:0x401a40 period: 2809 addr: 0x7fe2d13a1000
```

Intel uncore PMU support

- Intel SNB-EP uncore PMU available since 3.5
  - check with `/sys/devices/uncore_cbox_0`
- Intel IVB-EP uncore PMU available since 3.9
  - specs available now
  - libpfm4 support coming soon
- perf support: nothing specific

Hardware bugs

- LateGO (SNB): PEBS memory events may undercount
  - publicly documented with software workaround
  - L3 latency penalty w/ workaround: not integrated in kernel/perf tool
  - Use Andi Kleen's `pmu-tools` script to enable workaround (`latego.py`)

- HT bug (SNB,IVB,HSW): cross-thread counter corruption
  - publicly documented, no firmware workaround

```
$ toplev.py -l1 noploop 1
WARNING: HT enabled
Measuring multiple processes/threads on the same core may not be
reliable.
perf stat -x, -e '{r2c2,r100030d,r19c,r10e,cycles}' noploop 1
```

References

- IA-32 Software Developers Manual (SDM) Vol3b September 2013
- Andi Kleen's pmu-tools on github
- Intel SandyBridge-EP uncore PMU specs
- Intel IvyBridge-EP uncore PMU specs
- Intel SandyBridge specification update
- Intel IvyBridge specification update
- Intel Haswell specification update

Google

Haswell PEBS DLA Load events	Haswell PEBS DLA store events
MEM_UOPS_RETIRED.STLB_MISS_LOADS	MEM_UOPS_RETIRED.STLB_MISS_STORES
MEM_UOPS_RETIRED.LOCK_LOADS	MEM_UOPS_RETIRED.LOCK_STORES
MEM_UOPS_RETIRED.SPLIT_LOADS	MEM_UOPS_RETIRED.SPLIT_STORES
MEM_UOPS_RETIRED.ALL_LOADS	MEM_UOPS_RETIRED.ALL_STORES
MEM_LOAD_UOPS_RETIRED.L1_HIT	MEM_LOAD_UOPS_RETIRED.L2_HIT
MEM_LOAD_UOPS_RETIRED.LLC_HIT	MEM_LOAD_UOPS_RETIRED.L1_MISS
MEM_LOAD_UOPS_RETIRED.L2_MISS	MEM_LOAD_UOPS_RETIRED.LLC_MISS
MEM_LOAD_UOPS_RETIRED.HIT_LFB	MEM_LOAD_UOPS_LLC_HIT_RETIRED.XSNP_MISS
MEM_LOAD_UOPS_LLC_HIT_RETIRED.XSNP_HIT	MEM_LOAD_UOPS_LLC_HIT_RETIRED.XSNP_HITM
UOPS_RETIRED.ALL	MEM_LOAD_UOPS_MISC_RETIRED.UC
MEM_LOAD_UOPS_LLC_HIT_RETIRED.XSNP_NONE	MEM_LOAD_UOPS_LLC_MISS_RETIRED.LOCAL_DRAM
MEM_LOAD_UOPS_LLC_MISS_RETIRED.LOCAL_DRAM_SNP_HIT	MEM_LOAD_UOPS_LLC_MISS_RETIRED.REMOTE_DRAM
MEM_LOAD_UOPS_LLC_MISS_RETIRED.REMOTE_DRAM_SNP_HIT	MEM_LOAD_UOPS_LLC_MISS_RETIRED.REMOTE_HITM
MEM_LOAD_UOPS_LLC_MISS_RETIRED.REMOTE_FWD	MEM_LOAD_UOPS_MISC_RETIRED.NON_DRAM
MEM_LOAD_UOPS_MISC_RETIRED.LLC_MISS	